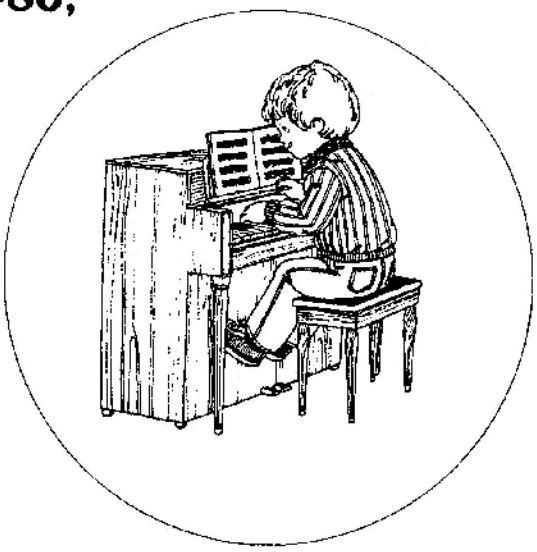
THE MISOSYS QUARTERLY

In this issue:

Corrected address:
Gary Lee Phillips
18404 Garden Valley Rd.
Marengo, IL 60152-9436 USA
tivo.over@gmail.com



MISOSYS will be closed December 22nd through January 2nd

Volume V.i **$10** **Fall 1990**

MIDI Your TRS-80

Gary Lee Phillips
4889 North Hermitage
Chicago, IL 60640
Telephone (312) 769-2988

So you'd like to get into computerized MIDI, but you already own a computer and there's no commercial interface or software for your favorite machine? If that's been holding you back, this project is for you. Although the parts are not exactly dime-store items, you can get them by mail order for less than $50 (see references for supplier list) and no fancy hardware knowledge is required. You will need to be able to use a soldering iron or a wire-wrap tool, and read a circuit diagram. A digital or analog multi-function meter is necessary, and a logic probe or oscilloscope will help if you have to debug your board after building it.

This hardware interface is designed to provide IMA standard MIDI in, out, and thru for a Radio Shack TRS-80 model 3, 4, 4D, or 4P. Other Z-80 based systems, including the model 1, 2, and 12, Kaypro Z-80 machines, Heath/Zenith Z-80 equipment, and S-100 systems should be able to use this interface after some adaptation of the bus interface portions.

What about software? No commercial software is available, but you can write your own simple programs from scratch in assembly language, C, or another compiled language (Fortran anyone?) and I will provide a reference for some excellent C code that can be adapted without too much trouble. Sure, your TRS-80 won't generate flashy graphics and colored windows, but it can function as an efficient patch librarian and a decent multitrack sequencer if you are able to work up the right software combination from sample routines in C and the assembly language interrupt module provided here.

Construction Details

I chose LS-TTL devices because they are readily available, inexpensive, and use relatively little power. The circuit could be quickly adapted to high-speed CMOS if you are familiar with the requirements of the latter. The only chip used that is difficult to obtain is the PC-900 optocoupler. I bought several from the Sequential Circuits parts department, since I own a Six-Trak. Check with keyboard repair operations in your area or call the manufacturer of your MIDI instruments for optocoupler availability. With appropriate pin changes, other chips may work in place of the PC-900.

You will need a source of steady DC current at about 9 volts and 400 milliamps to power the interface. I used a surplus power adapter from a calculator. The onboard 7805 regulator adjusts the power to exactly 5 volts for the TTL circuitry.

I used wire-wrap construction on a piece of experimenter's perfboard to build my prototype, and after initial debugging I soldered the wires in place. It looks a little funky, but it's still working fine after two years. You can use whatever construction methods you feel comfortable with, but make sure you do a neat job, and keep the length of connecting wires as short as possible. Use sockets for the chips so you can replace them or rewire any mistakes without handling the IC's themselves.

Build the power regulator section of the circuit first, and test it before continuing. A simple voltmeter reading is adequate. You should see a steady voltage at the 7805 output pin between 4.9 and 5.1 volts DC. If the reading varies from this level, try another 7805 and/or check your wiring. Don't feed incorrect voltages to your TTL chips, or you will destroy them. Connect a 220 ohm 1/2 watt resistor from the 7805 output to ground, and check the voltage again. Still 5 volts? Good, it's safe to proceed.

Use wire-wrap sockets with long pins, and attach them to the perfboard with a drop of glue. If your experimenter's board has printed circuit bus lines for the power supply, wire these and use them. If not, position several soldering pins along the middle of the board for the +5 and ground connections, and solder these to a supply wire, preferably no. 18 or larger. Color-coded insulation is helpful but not necessary. Just make sure you can tell which supply points are live and which ones are grounded, so you don't connect any chips backwards. If you make that mistake, after the smoke clears you'll have to start all over.

Install several 0.1 microfarad disk capacitors along the power bus, between the +5 volt line and the ground. These bypass caps will help absorb the spikes created when the gate voltages switch rapidly during operation.

Now install the IC sockets and remaining parts. Make each connection carefully, then double check it for correct location. Finally, check it off on the diagram with a small tick. Connections to +5 volts or ground are indicated in table 1 and most are omitted from the diagram to keep the picture simple. After all the wiring is complete, and you are sure it is correct, power up the board again and test the voltage at each socket. If it still shows a steady +5 volts on each supply pin, you are ready to install the chips.

You can use a grounding wrist strap from Radio Shack for this operation, or just work on a sheet of foil or a cookie sheet. Avoid static electricity, which can damage the chips. Don't work in a carpeted room or under very dry conditions if you can avoid it. Be careful not to bend pins, and be sure to match pin one of each chip to pin one of the socket. The end of the chip with pin one is usually marked with a white dot, or has a curved notch in it.

An interface cable is required to connect the board to the I/O bus (the one normally

used for a hard disk drive) on the TRS-80. If you have a hard disk installed and want to use it for program and data storage with the MIDI interface, you will need a "Y-connector" for the port, available from Alpha Products (see references).

By using a male 50-pin dual header on the interface board, I was able to wire a straight-through cable from 50-conductor flat ribbon cable, using a female dual header on one end, and a 50-conductor female card edge connector on the other end. Shorter lengths (under 36 inches or so) will be most reliable. Radio Shack sells a cable intended for their 5-megabyte hard disk that can be used, but you can build your own for less money.

After chips are installed, and you check once again for accuracy, apply power to the board. If there's no smoke, and nothing gets hot (warm is OK), you are ready for a functional test. Turn power off, and attach the interface cable to the 50-pin header. Attach the other end of the cable to the 50-pin card edge of your computer (with the computer turned off, please). Be sure you orient the card edge connector correctly, pin one to pin one. Each model of TRS-80 is different in this respect, so check your technical manual for accuracy. Power up the interface first, and if no immediate problem appears, turn on the computer. If the computer does not initialize normally, shut it off immediately, and recheck all connections to the bus at both ends of the cable. The computer does work when the interface is not attached, right? Try it to be sure, and then examine your cable and board for bad connections or shorts.

Once you get both the computer and the interface operating without apparent problems, turn off the power again and add MIDI cables between the interface and your keyboard. MIDI In on the interface connects to MIDI Out on the synth and vice-versa, OK?

Enter the Basic test program from listing 1, and run it. If everything has gone well, you will hear the synth play a scale and then stop. It worked? Good, that means that the output circuitry and bus interface are functioning. If it didn't do anything, or you got strange results ("stuck" notes, weird noises) then immediately turn everything off and recheck all steps up to this point. If you can't find the problem, proceed to the debugging discussion at the end of this article.

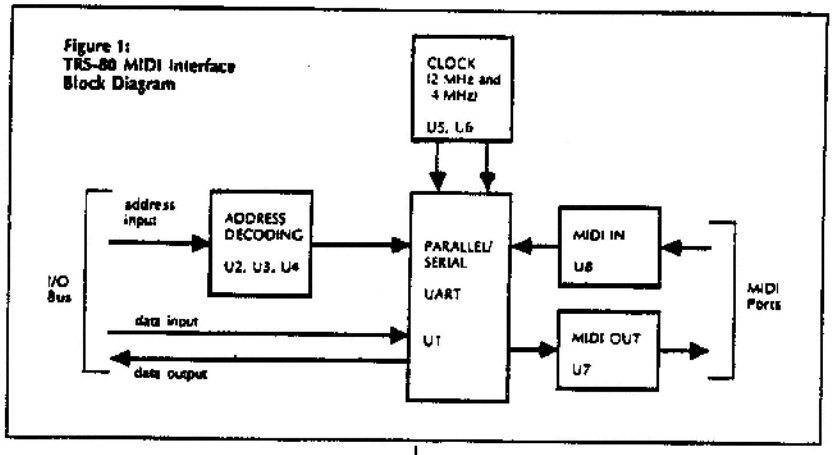


Figure 1:
TRS-80 MIDI Interface
Block Diagram

Testing the MIDI In port is more difficult. Basic isn't fast enough to trap the codes generated by key presses on your synth unless it is assisted by a routine in assembly language. If you have an assembler, you can assemble the small test routine in listing 2 and run it. Press a few keys slowly (no, this won't handle any fast riffs) and you should see the MIDI codes appear on your screen. If that works, you are ready for software development.

How it Works

The block diagram in figure 1 shows the four main subassemblies of the interface. The Z-80 bus is a parallel bus. That means that all eight data bits or eight port address bits are transferred simultaneously over eight separate conductors. MIDI is a serial communication system, in which each bit is transferred sequentially over the same wire. The Z-80 SIO chip at the heart of the interface performs translations from the computer's parallel bus to MIDI's serial bus and back. The address decoding circuitry keeps the interface from responding to I/O requests destined for other internal or external devices.

The clock uses a crystal to generate square waves at 4 MHz to drive the SIO. A dual flip-flop (U6) divides this frequency by two, generating a 2 MHz signal to set the appropriate baud rate for MIDI. The SIO will internally divide this frequency by 64 when it is properly initialized by the software, thus yielding the standard 31.25 kilobaud rate of MIDI.

Power supply regulation by the 7805 guarantees a steady 5 volts DC to power the other parts of the interface. By feeding 9-12 volts into the 7805, we provide a little extra protection against undervoltage if the AC line voltage droops.

Writing the Software

This is the hard part of the project, I'm afraid. If all you know is Basic, you're going to have to do some studying. If you already know C and have a good compiler, you're practically home free. You can write MIDI applications for the TRS-80 in any language that is fast enough to keep up with your intended operations.

Suitable C compilers are available from MISOSYS or Manx Software Systems. The Microsoft Fortran available through Radio Shack also works very well if you like Fortran. Turbo Pascal is suitable, if you run CP/M rather than TRSDOS, and some sample code by Donald Swearingen is available (see references).

Jay Kubicky's article in *Byte* (again, see references) has complete C source for a simple sequencer. A patch librarian does not demand as much speed as an eight track sequencer, and will be easier to design and write, as well. If you must use Basic, get a Basic compiler because interpreted Basic just isn't fast enough on these machines.

The SIO control and data ports are mapped as shown in table 2. Channel A is the MIDI In channel, and also the MIDI Out channel. If the jumper is installed for alternate MIDI Out, then Channel B can be used to provide a totally separate MIDI output to the MIDI Thru connector. With the jumper in the position shown on figure 2, however, Channel B cannot be used at all and the MIDI Thru connector will "echo" whatever is received at the MIDI In port. If you wish to change the base port assignment to something other than port address 64, you can do so by rewiring the sections of the 74LS04 (U2) appropriately. The address I used is not in conflict with any internal or typical external TRS-80 device of which I am aware.

Listing 3 is a simple device driver for TRSDOS or LS-DOS that provides unbuffered I/O with time-stamping of received data. This driver supports the same operations that are provided by the DOS standard serial port driver, including use of the "wake-up" vector to permit immediate buffering of input by your application program. See the technical documentation for your DOS version for details on use of standard device I/O.

The SIO can be programmed using the standard techniques described by Coffree or Zilog (see references). This sample driver merely presents one of many viable approaches.

Patch librarians do not generally require timing information, but sequencers do. The internal clock interrupt of the TRS-80, running as it does at 60 Hz, can be used to time input and output information for several tracks in a sequencer. Software timers driven by this interrupt can be linked through the task controller of the LDOS, TRSDOS, or LS-DOS operating systems. Alternatively, a Z-80 CTC or similar independent timer could be added to the interface design.

If you have a model 4 computer, by far the most common TRS-80 these days, you may also wish to look into a speed-up and memory expansion board. I use the XLR8er board from H. I. Tech, which gives the equivalent of an 8 MHz CPU speed and an added 256K of memory with a fast RAMdisk driver. The XLR8er has a Z-80 compatible CPU replacement, the Hitachi HD64180, which includes additional serial ports and two built-in timers that can be used in your MIDI software.

While the TRS-80 may not be suitable for high-powered professional studios or stage performers, it is entirely adequate for use by any intermediate or advanced amateur or hobbyist in the MIDI music field. Developing your own software takes a while, but when you finish you will have a more thorough understanding of the internals of MIDI and the hardware underpinnings of your music. After all, Wendy Carlos designs her own hardware and writes her own software, and still finds time to create beautiful musical compositions, so why should you feel forced to buy canned software and commercial hardware? The feeling of satisfaction at creating your own interface and working software can be very rewarding in its own right.

Table 1: IC Power Connections

	+5 volts	Ground
U1 Z-80A SIO	pin 9	pin 31
U2 74LS04	pin 14	pins 7,13
U3 74LS30	pins 11,12,14	pin 7
U4 74LS32	pins 1,4,12,14	pin 7
U5 74LS14	pin 14	pins 7,5,9
U6 74LS73	pins 4,7,10	pin 11
U7 74LS05	pin 14	pins 5,7,9
U8 PC-900	pin 6	pin 5

Table 2: Port Address Decoding

Address	Port Identification
64 (40H)	SIO Channel A data
65 (41H)	SIO Channel B data
66 (42H)	SIO Channel A control/status
67 (43H)	SIO Channel B control/status

Debugging Clinic

So you built it but it doesn't work? This may sound monotonous, but the first thing to do is go back over the instructions and check every step. Be absolutely sure your circuit matches the diagram.

Use the voltmeter to check supply voltages at each chip. Test the interface cable for shorts or open circuits. Is your computer working correctly? Are you sure the synthesizer is enabled to accept MIDI input? (Some equipment requires a console command sequence before it "listens" to MIDI commands. Check your manuals.)

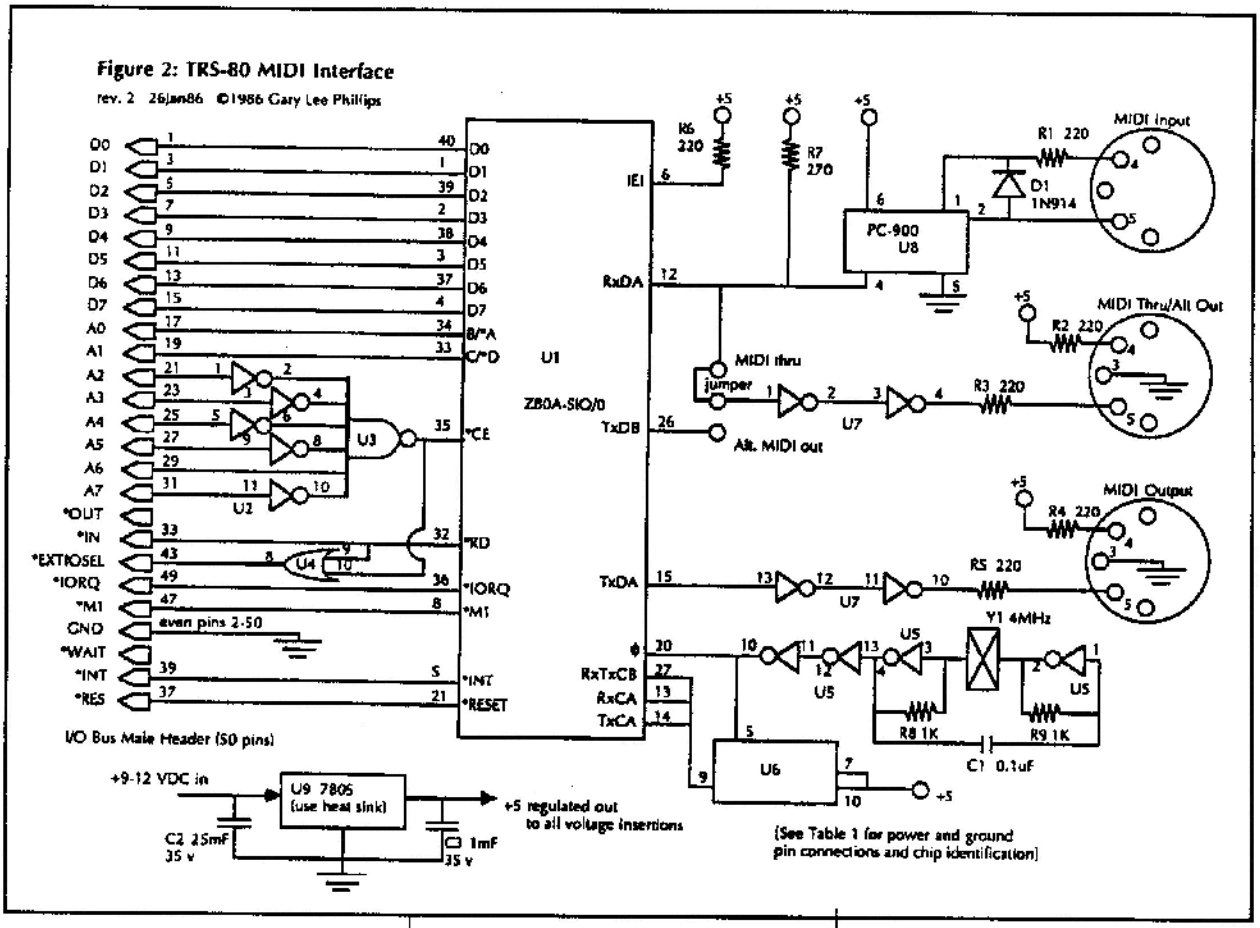
If it still doesn't work, you are going to need a logic probe or an oscilloscope to check things out. Get help from someone with a little experience if you don't know how to use these tools.

Make sure the 4 MHz clock is cycling on your interface. Then be sure the divide-by-two flip-flop is supplying 2 MHz to the Z-80 SIO.

Test the bus interface on your computer by attaching some other known working device, such as an Orchestra 90. If MIDI output works, but not input, check the *EXTIOSEL signal to be sure it is being pulled low when input is requested from the SIO.

If replacement chips are available, try swapping them. You can occasionally get a bad chip from a dealer, or may have bent or broken a pin while installing it.

Above all, persevere. I had some trouble getting the 4 MHz clock working at first, but replacing the parallel capacitor did the job. Other than that, it worked right from the beginning, and I'm not an advanced technician or designer.



References: Parts suppliers:

Alpha Products, 242-B West Avenue, Darien, CT 06820 — bus cables and interfaces.

Radio Shack, 1 Tandy Center, Ft. Worth, TX 76102: perfboards, tools, capacitors, connectors, sockets, resistors, cables.

JDR Microdevices, 2233 Branham Lane, San Jose, CA 95124, 800-538-5000: stocks all IC's and parts except the PC-900.

Jameco Electronics, 1355 Shoreway Rd., Belmont, CA 94002, 415-592-8097: stocks all IC's and parts except the PC-900.

References: Printed data:

Coffree, J. *Z80 Applications*, Sybex, 1983. Good explanation on use of Z-80 SIO for serial I/O.

Kubicky, J. "A MIDI Project." *Byte*, June, 1986, p.199. Programs available on diskette from *Byte* Listings.

Soltoff, R. *The Programmer's Guide to LDOS/TRSDOS Version 6*, 1983. Available from MISOSYS, Inc. (see below)

Swearingen, D. "MIDI Programming." *Byte*, June, 1986, p.211. Programs available on diskette from *Byte* Listings.

Zilog, Inc. *Z-80 SIO Technical Manual*, 1978.

References: Compiler vendors:

Borland International, 4585 Scotts Valley Dr., Scotts Valley, CA 95066—Turbo Pascal for CP/M-80 operating system.

Manx Software Systems, 1 Industrial Way, Eatontown, NJ 07724 — Aztec C for CP/M-80 and TRSDOS operating systems.

MISOSYS, Inc., P.O. Box 239, Sterling, VA 22170 — C, RATFOR, macro assembler for TRSDOS. Also sells the XLR8er enhancement board.

Radio Shack, 1 Tandy Center, Ft. Worth, TX 76102—Microsoft Fortran for TRSDOS, model 4 version includes macro assembler.

```
Listing 1
10 REM - Sample program to test MIDI output
20 REM - Should play a major scale on C
30 REM - Copyright 1988 Gary Lee Phillips
40 REM - First initialize the interface
50 REM - Polled mode, no interrupts
60 A% = 64 : REM - Set base address
70 RESTORE
80 FOR I% = 1 TO 11
90 READ N% : OUT A% + 2,N% : OUT A% + 3,N%
100 NEXT I%
110 REM : Now send commands to play 8 notes
120 C% = 3 : REM - Set base channel (1-16)
130 FOR I% = 1 TO 8
140 REM - Turn on a note
150 READ N%
160 OUT A%,143 + C% : REM - note on
170 OUT A%,N%       : REM - pitch
180 OUT A%,64       : REM - velocity
190 REM - Let note sound for a while
200 FOR J% = 1 TO 300 : NEXT J%
210 REM - Turn same note off
220 OUT A%,128 + C% : REM - note off
230 OUT A%,N%       : REM - pitch
240 OUT A%,64       : REM - velocity
250 REM - Wait a while before next note
260 FOR J% = 1 TO 75 : NEXT J%
270 NEXT I% : REM - Play next note
280 END
290 REM - Initialization code values
300 DATA 24,20,196,19,193,21,104,18,255,17,0
310 REM - Major scale pitches
320 DATA 48,50,52,53,55,57,59,60
```

```
Listing 2
;       TESTINP - brief test for SIO MIDI input
;       Be sure synthesizer MIDI output from keyboard
;       is enabled. Assemble and run program, and
;       MIDI codes will be displayed on screen as
;       keys are pressed or released.
;       Copyright 1988 Gary Lee Phillips
        ORG     3000H
TESTINP LD      A,69H           ;Clear screen
        RST     28H
        LD      HL,SINON
        LD      A,0AH           ;Display message
        RST     28H
; Initialize for polled mode receive
        LD      B,ICNT          ;No. of bytes
        LD      HL,IDAT         ;Initialization data
INIT    LD      A,(HL)          ;Send init codes
        OUT     (42H),A
        OUT     (43H),A
        INC     HL              ;Point to next
        DJNZ    INIT            ;Loop till done
TEST    LD      A,6AH           ;Test Break key
        RST     28H
        JR      Z,READ
        LD      A,16H           ;Normal exit
        RST     28H
READ    IN      A,(42H)         ;Get status
        AND     01H             ;Data ready?
        JR      Z,TEST          ;No, try again
        IN      A,(40H)         ;Yes, read it
        LD      C,A
        LD      HL,BUFF
        PUSH    HL
        LD      A,62H           ;Convert to hex
        RST     28H
        POP     HL
        LD      A,0AH           ;Display value
        RST     28H
        JR      READ            ;Try next
SINON   DB      'MIDI Input Test.
        DB      'BREAK to quit.',0AH,0DH
BUFF    DB      '  ',20H,03H
IDAT    DB      18H,14H,0C4H,19H,0C1H
        DB      15H,68H,12H,0FFH,11H,00H
ICNT    EQU     $-IDAT
        END     TESTINP
```